WaveSorter

A free, open-source offline spike sorter

Version 1.0

Matt Phillips, mattphillipsphd@gmail.com

Nov. 2, 2012

Table of Contents

0. Notable, Non-Obvious Features of WaveSorter
1. Introduction, Basic Procedure for Using WaveSorter
2. Getting Started
3. The Main Window
4. The WaveSpace Window
5. Cluster Windows
6. Validation
7. File Formats
8. Citation and License

0. Notable, Non-Obvious Features of WaveSorter

This section is for people who are inclined to just play around with the software and get working without really bothering to read the manual. Most functionality of WaveSorter can be extracted by going this route, but there are some things that might be obscure or overlooked unless noted:

1. The 'D' in Raw-D, Slope-D etc. refers to 'differentiated'. In these cases the transforms were carried out on the differentiated waveform. (Thanks to Mark Churchland for this suggestion.) In many situations the differentiated waveforms exhibit substantially different between-cluster differences than the raw waveform versions.

2. You can select a subset of all the samples from a transform to use in an automatic clustering algorithm. Most samples in a given transform carry virtually no usable information with respect to clustering—e.g. With PCA that's by design. Use the scroll wheel to zoom in and out, and the mouse to drag the waveforms to the left or right so as to bring just the desired samples into the Waves window. Then click 'Set Samples' and only those samples that were visible will be used. These samples will be surrounded by a blue box.

3. When zooming in Axis mode, double click anywhere on the Histogram window to revert to the original dimensions.

4. After manual clustering, the transform sample displayed in the histogram will be reset to 0. You can jump to the location of Draw Objects you have created by clicking on their IDs in the drop-down list beneath the histogram.

5. Centers, in manual clustering, partition the set of otherwise unaccepted or not explicitly (via a polyline) rejected waveforms into cells, each of which contains the waveforms nearest to that center. These let the user divide the group of unaccepted waveforms approximately into clusters, which can improve the statistical validation.

6. WaveSorter is highly parallelized, in terms of GUI operations as well as mathematical computations. If you are sorting large (100,000s) files and were looking for an excuse to upgrade to a machine with more cores, this could be it. Processing time is reduced almost exactly proportional to the number of cores.

7. WaveSorter runs very naturally over a server, as the facility to create individual account profiles is built in. Given WaveSorter's high degree of parallelization, if an available server is much more powerful than your personal computer it might make things faster to do them on the server. Remote desktop (X11 forwarding for Mac/Linux, PuTTY/XMing for Windows) on an intranet can be quite fast, not inducing significant lags.

1. Introduction and Basic Idea

Introduction:
As electrophysiological recording technology and analytical techniques advance, easy-to-use, powerful spike sorting tools become important for two reasons:

- They allow the extraction of meaningful waveform metrics, in order to help identify different neural subtypes. (e.g.Mitchell et al. *Neuron* 2007)

- The increasing number of channels used in electrophysiological preparations increases the need for post-hoc validation of online spike discrimination.

WaveSorter distinguishes itself in the extent to which it allows the user to dynamically visualize his/her data, the wealth of clustering options it offers the user, and the speed at which it operates.

Not kidding about speed, by the way. WaveSorter will **max out** your machine. If you have a powerful (8+ cores) computer and you are clustering large files, or batch processing, make sure there's sufficient ventilation so that your computer doesn't overheat.

Basic Idea:
WaveSorter takes as inputs data files in which the potential waveforms have already been identified, and are stored as arrays of numbers (e.g. voltage levels) of a fixed size. WaveSorter does not itself do this initial extraction from the raw signal, which in typical setups is accomplished online.

Once you've loaded a file, WaveSorter computes various transforms (PCA, Wavelet, et al.) and metrics on the set of waveforms. These transforms enable the user to see structure in the data—particularly, the separation of waveforms into separate groups (clusters) that would be difficult to see inspecting just the raw waveforms. In general, these transforms accomplish a form of dimensionality reduction, enabling the user to focus in on the real sources of variance in the data.

As the transforms are being computed, the user can visually inspect the data in a dynamical, graphical way to find that structure. Once he/she has identified the crucial dimensions along which the waveforms vary most, the user can then apply either automated or manual clustering algorithms to create the actual clusters.

Finally, the user can save the clustered data in any of several data formats to use in subsequent data analysis.

2. Getting Started

WaveSorter is intended to be ready to go as soon as you download (sourceforge.net/projects/wavesorter) the relevant version and extract it. In addition to the 3 major 64-bit platforms, WaveSorter will also work on 32-bit platforms but in that case you will have to compile the code yourself, as well as finding and installing the external libraries upon which it depends (Qt and the GSL).

A relatively painless install is a critical first step to a good user experience, so if it's not working for you please email me (mattphillipsphd@gmail.com) and I'll work with you to get it up and running. One thing to try yourself though on Mac and Linux is to open a terminal, `cd` into the WaveSorter-*-1.0.0 folder and execute this command:

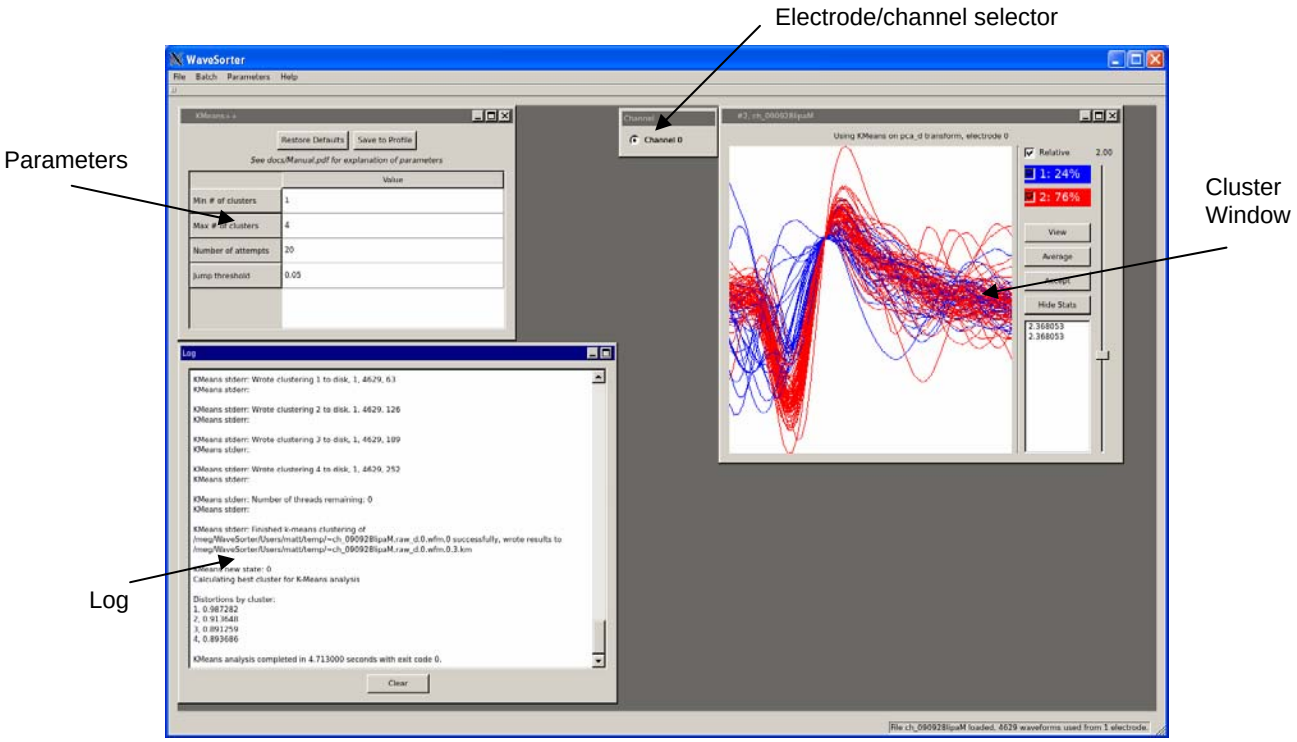
```
$ chmod u+x WaveSorter
```

Then try to run it with `./WaveSorter`. This should work; at the very least this will give you an informative error message. In general, if it doesn't launch it's because either a) there's a low-level platform incompatibility, in which case the program will have to be recompiled, or b) WaveSorter can't find the libraries it needs. In the latter case you may be able to find and install these (Qt and GSL) relatively easily. In any case, as long as you're on one of the major platforms and your system is reasonably recent there will be a way.

When you do launch WaveSorter successfully, two windows will appear; the larger main window, and a small dialog box entitled 'Choose Profile'. The first time you use WaveSorter, click 'new'; you can then choose a profile name, after which you will be prompted for a) the default directory whence you would like to pull your waveform files, and b) the output directory, into which WaveSorter will write its clustering files. These can be the same directory.

That's all there is to the install and setup, WaveSorter is now ready to go.

3. The Main Window



This holds and organizes all of the windows that the user interacts with during the course of a session. In addition to clusterings, this window also displays:

1. The log, which displays all system messages, warnings and errors, as well as the output from the external clustering algorithms as they progress.
2. A small gui indicating the different channels (electrodes) contained in the currently selected session. Selecting a channel will load the corresponding waveforms into the WaveSpace window.
3. Parameters. Use these to parameterize both the external clustering algorithms as well as the session as a whole.
4. Clustering Windows. See section 6.

Menu options:

File:

File->Load Waveform File: Self-explanatory. If the file has an online clustering available, this will be displayed as soon as the file loads.

File->Load Classification: This loads a clustering (*.clu.*) file that you have previously generated. In order to do this the file from which the clustering was generated must currently be in WaveSpace.

File->Save Spikes:

File->Save Spikes->Into REX E-File: See File Formats section for more information.

File->Save Spikes->Cluster file: Saves an ASCII text file (*.clu.* file). See File Formats section for more information.

File->Save Spikes->Simple Binary: Saves a binary file (*.bin.* file), storing only whether a waveform was accepted or not. See File Formats section for more information.
File->Save Spikes->Simple ASCII: Same thing as Simple Binary except that the output is in ASCII format (*.ascii.* file). See File Formats section for more information.
File->Save Spikes->Save All: This creates a tab-separated text file (*.data.* file) . See File Formats section for more information.

Batch:
Batch->Run: Prompts the user to select a directory, verifies that which transforms will be computed for each file, and then starts the batch processing. Note that the clustering method and parameters are taken from whatever is in the WaveSpace window at the time; for this reason the Batch option is grayed out until the first file has been loaded. You may continue to use WaveSorter to analyze other sessions while batch processing is taking place. Clusterings will appear in the main window as they are completed.

Parameters:
Parameters->Session: Note: These parameters must in general be saved to the profile in order to be in effect when a new file is loaded.

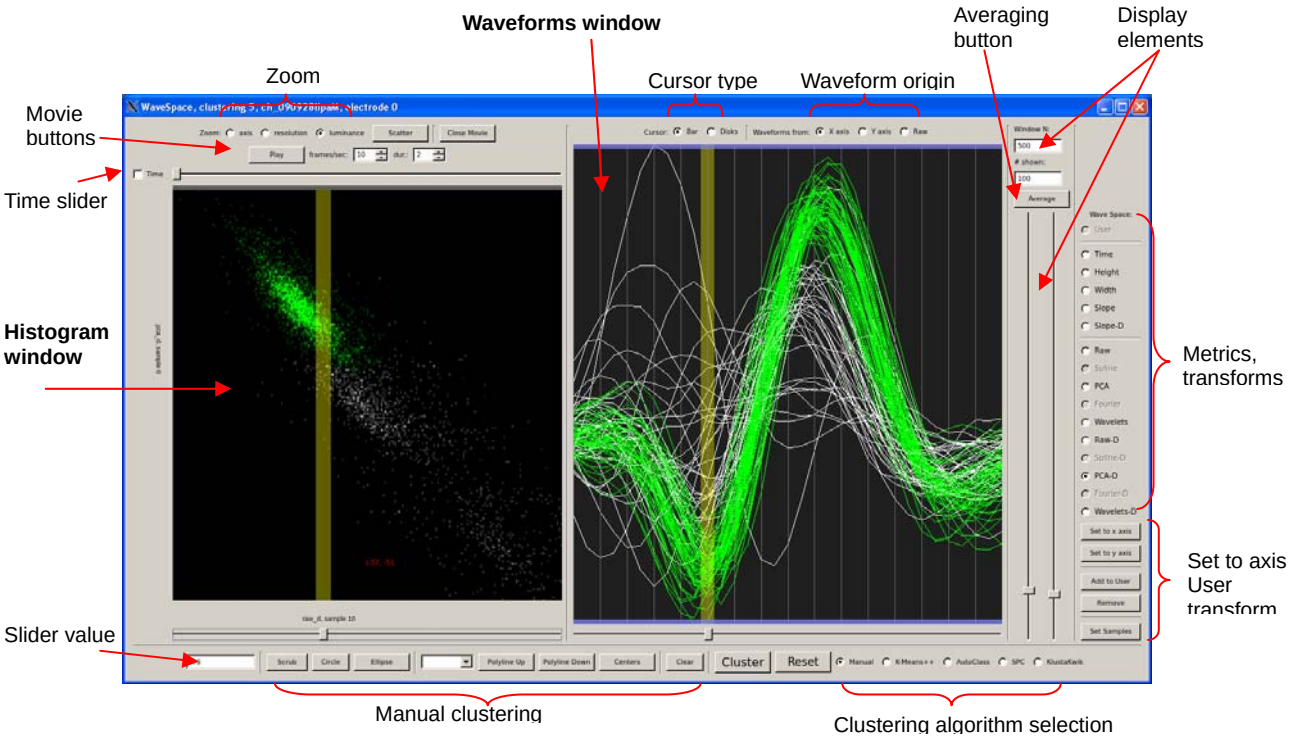
- Data Directory: Default directory when loading data.
- WS Data directory: Directory where saved data (including parameter files) will be put.
- Align waves: {**none**, peak, trough} How to align the waveforms when the session is read. Note that choosing peak or trough will increase the load time substantially.
- Scrub threshold (SDs): During loading the height (max – min) of every waveform is calculated and the mean is found. Waveforms whose height is greater than *val* SDs away from the mean will be discarded from the analysis.
- Num Spline Breakpoints: For the Spline transformations. Breakpoints are evenly spaced across the sample array, and include the beginning and end.
- Num Batch Threads: Currently not in use.
- Make *: {yes, no} Whether or not to generate this transform when a file is loaded. If you know you won't use it (e.g., the Fourier transforms), then you can avoid wasting the CPU required to compute it; this is especially relevant for batch processing.

Parameters->*Clustering algorithm*: Sorry, you're on your own here. The meaning of some is obvious, like 'MinClusters'; but in general you'll need to look at the documentation or source code comments in the algorithms themselves for explanations. I note that the default parameter values for Kmeans++ and Super-Paramagnetic Clustering seem to work pretty well.

Help:
Help->About: Standard credits, attributions, contact info etc.

4. The WaveSpace window

This is where virtually all of the data manipulation and visualization is done. The best way to learn about its functionality is to load a file and just play around with all the buttons and sliders.



Waveforms window:
Displays a subset of the waveforms currently selected in the **histogram window** by the bar or disks. A proper subset is displayed when the number selected in the histogram is larger than the value of the "# shown" edit box; in this case, every Nth waveform is selected so that the resulting group contains that value. Waveforms from the transforms are displayed when they are set to the x or y axis. Waveform 'slices' for plotting in the **histogram window** are selected using the slider at the bottom, which will move the yellow scanline accordingly. Using the mouse scroll wheel and buttons, it is possible to zoom in and drag the waveforms to the left or right, especially valuable when you want to **Set Samples**. No waveforms are displayed when metrics are set to an axis, since they are 1-dimensional. The histogram axis whose waveforms are displayed can be chosen with the **waveform origin radio buttons**; or, regardless of histogram axes, raw waveforms can be displayed.

Waveform origin radio buttons:
These determine which axis will supply the transform whose waveforms are viewed within the **waveforms window**. Alternately, if 'Raw' is selected, the raw waveforms can be used regardless of which transforms are plotted in the **histogram window**. The value of these buttons is set automatically whenever the user sets a transform to an axis with the **set axis buttons**.

Display elements:
These control the width of the bar (x axis or time)--'Window N' and corresponding slider--and the

number of elements shown in the **waveforms window**, '# shown' and corresponding slider.

Averaging button:
Toggles between showing individual waveforms in **waveforms window** and showing the average waveform (+/- 1 SD) for the selected waveforms. All non-accepted clusters are averaged separately.

Histogram window:
This displays a heat map (2-D histogram) or scatter plot of 'slices' through the different transforms, or metrics, chosen in the **waveforms window**. After a clustering has been accepted, with the 'Accept' button on a cluster window in the main gui, accepted waveforms will be shown in green and unaccepted waveforms will be shown in white. Using the **zoom radio buttons** it is possible to change the size, position, brightness, or resolution of the histogram. (Note: When the axis is being zoomed, double-clicking the mouse will return it to its original dimensions.) Using the **cursor radio buttons**, choose either the bar, moved using the slider at the bottom, or the disks, which are dragged and dropped, to select particular waveforms for viewing in the **waveforms window**. All manual clustering elements, controlled with the **manual clustering buttons**, are 'drawn' with the mouse directly on the histogram.

Zoom radio buttons:
These allow the user to use the scroll wheel to increase or decrease different properties of the histogram in the **histogram window**, and in the case of axis zoom, the user is also able to reposition the axis. 'Axis' corresponds to a standard spatial zoom. 'Resolution' changes the bin size of the 2D histogram. 'Luminance' changes the brightness of the histogram, which is useful especially on monitors with a low dynamic range. The last two zoom options are not operative in scatterplot mode.

Cursor radio buttons:
Choose which cursor will be used in the **histogram window** for selection of waveforms that are viewed in the **waveforms window**. The bar is moved using the slider at the bottom of the **histogram window**. The disks appear in the upper left-hand corner when first selected, and are to be dragged and dropped. Note, they can be resized with the scroll wheel.

Movie buttons:
WaveSorter offers the user the ability to watch the session unfold either in time or as a function of the value of the x axis. This can be particularly useful for visual inspection of a clustering. Click 'Movie' and then a panel will appear, where the user can start and stop the movie and change how fast it plays. When the movie progresses in time, the yellow box beneath the **time slider** will indicate the point in the session whence the waveforms are drawn. When the movie progresses through the x-axis, the bar will do this. 'Bar' must be selected among the **cursor radio buttons** for the movie to work in this case.

Time slider:
This is enabled/disabled by clicking the 'Time' checkbox. When enabled, a yellow rectangle will appear at the top of the **histogram window**, and only that subset of waveforms occurring during that interval in the session will be plotted. If 'Bar' is selected from the **cursor radio buttons** all waveforms from that interval will be displayed in the **waveforms window**; if 'Disks' is selected, only those waveforms selected by a disk will be.

Slider edit box:

Shows the current value of the histogram slider.

Clustering algorithm selection radio buttons:
Whichever of these is selected determines which clustering algorithm will be used the next time a clustering is launched. When 'Manual' is selected, the **manual clustering buttons** will be displayed.

Manual clustering buttons:
These buttons provide several different techniques, to be used together or separately, for manually clustering the waveforms. **Note:** When the **time slider** is enabled, Draw objects will accept/reject *only those waveforms occurring during the specified interval*. The duration of the Draw Object is shown in green (Circles, Ellipses) or red (Polylines) beneath the slider. This feature is very useful for clustering when the waveform shape of a neuron changes over time due to e.g. changes in isolation.

- Scrub: Lets you select individual waveforms with the mouse (highlighted in the **waveforms window**) and toggle them between accepted/unaccepted.
- Circle: Manually draw a circle around a group of waveforms you want to accept. You do not need to close the circle yourself, when you finish the program will automatically connect the first and last points.
- Ellipse: This draws an ellipse around a preselected group of points. These points are first selected by drawing a Circle around them. After clicking 'Finish Circle', the Circle will disappear and be replaced by an ellipse whose main axis lies along the axis of greatest variance of the cluster. A spin box will appear which lets the user adjust the size of the ellipse measured by SDs away from the center.
- Polylines Up, Down: These *exclude* waveforms in the specified direction relative to a line drawn with a mouse. These are especially useful when the x-axis is time, or otherwise to 'trim' a clustering. Note: You can create a line which extends horizontally from the mouse cursor to the nearest horizontal border by double-clicking.
- Centers: These partition the set of waveforms which have neither been explicitly accepted or rejected in to cells, defined for each waveform by the center nearest to it. These cells then become clusters once a clustering is launched. The motivation for this is to improve the quality of the validation statistics for manual clusters (see *Validation*).

There are two other pieces of functionality in this group: the 'Clear' button, which clears currently visible transforms (but does not reset the classification, if there is one), and the spin box. The spin box contains the 'location' (transform, sample number) of all Draw objects the user has created. Click on a location to view the Draw objects there.

Metrics, Transforms radio buttons:
Self-explanatory for the most part, select a transform and then click the desired **set to axis button** to plot it on the histogram. However, in addition to the precomputed transforms, the user is also able to generate his/her *own* transforms ('User' radio button) out of slices of the precomputed metrics and transforms. To do this, choose a slice, then click the 'Add to User' button from the **user transform buttons**. Continue to do this until you have built a custom transform to your liking. All slices are scaled to the range 0-1000 to give them roughly equal variance. If you have the transform displayed in the **waveforms window**, you can remove unwanted slices by clicking the 'Remove' button.

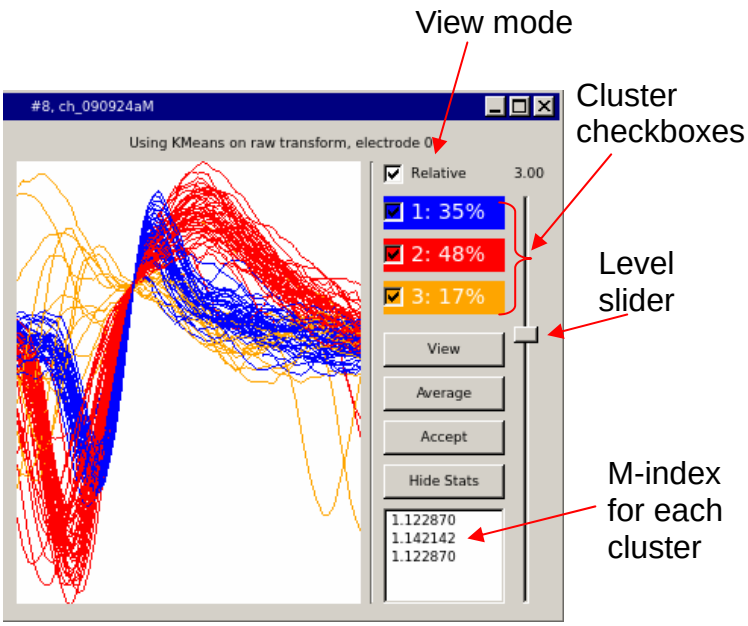
Set to axis buttons:
Self-explanatory; they set the specified **histogram window** axis to the transform that has been selected from among the **metrics, transforms radio buttons**, showing the slice corresponding to sample 0.

User transform buttons:
Add or remove transform slices to the current user transform.

Set Samples button:
For automatic clustering algorithms, click this button to use only samples within the range visible within the **waveforms window**. Once you do this, and zoom out, the selected samples will be surrounded by a blue box. Subselecting samples like this is a valuable technique for automatic sorting, as it can remove uninformative dimensions from the analysis (which would otherwise contribute noise) and also speed up clustering speed dramatically, depending on how small the subset is. Clicking on the **Reset** button resets these.

Cluster, Reset buttons:
Fairly self-explanatory; 'Cluster' starts a clustering (whose progress you can follow in the log), 'Reset' removes all Draw objects, resets the selected samples, and marks every waveform as unaccepted. Note that while a clustering is taking place you can still make full use of WaveSorter to do further analysis, even launching further clusterings.

5. Cluster Windows



Cluster windows appear in the Main MDI window whenever a clustering has completed, including clusterings that are read from a file. Once a clustering has been accepted and is represented in the WaveSpace window, the visible waveforms will correspond to the visible waveforms in the **waveforms window** there.

View mode checkbox:
If checked, the vertical extent of the waveforms will be scaled so that they exactly fit in the box (as is done in the **waveforms window** in the WaveSpace window). As the viewed waveforms change, so will the scaling. If unchecked, the vertical extend of the waveforms will be scaled so that all waveforms from the entire session will fit into the box. Moreover, in this case, the user can use the mouse to expand or contract the waveforms vertically (scroll wheel) or drag them up and down (mouse button). This mode is useful when you want to see how waveform shape evolves over time.

Cluster checkboxes:
These display the percent of the total waveforms constituted by the corresponding color. When checked, the waveforms in this cluster will be counted as accepted when the **Accept Button** is clicked. Multiple clusters can be accepted, analogous to 'merge' in other programs.

Level slider:
Two external clustering algorithms, Kmeans++ and Super-Paramagnetic Clustering, do not return a single clustering but rather a set of clusterings which varies as a function of some parameter. For Kmeans++, the number of clusters is that parameter. By default, it returns what counts as the 'best' clustering according to the Mahalanobis distance delta method described here:

http://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set#An_Information_Theoretic_Approach.5B7.5D

However by moving the slider the user is free to select whichever clustering seems right. The range of clusterings Kmeans++ considers can be controlled by the corresponding parameter. For Super-Paramagnetic Clustering, the parameter is 'Temperature'; the reader can follow the reference given in the About box for further explanation.

View button:
Loads the current clustering into the WaveSpace window.

Average button:
Like it's counterpart in the WaveSpace window, switches between individual and averaged waveforms.

Accept button:
Groups all waveforms from selected clusters as accepted and all others unaccepted, and loads this clustering into the WaveSpace window.

Stats button:
Shows or hides the M-index associated with each cluster in the current clustering. See *Validation* for more information.

6. Validation

Spike sorting is fundamentally a poorly-defined problem. We do not know a priori the actual variance of action potentials (measured along a particular dimension) generated by a particular neuron during a particular recording session. Relative electrode placement, cell type and filter settings affect what the measured 'true' variance would be. Thus ultimately, any algorithmic method of statistical validation of a clustering can only be assessed by whether its results 'look right'.

Nonetheless, where the aforementioned sources of uncontrolled variance are approximately constant across an experiment, it can be useful to put a number on the extent to which identified clusters seem well-defined and distinct from one another, and set a standard which a clustering has to exceed to be acceptable.

That's what's done here. The Dunn Index (http://en.wikipedia.org/wiki/Dunn_index) captures the intuition that in a good clustering, between-cluster variance is much greater than within-cluster variance; good clusters are tight and far apart. However the Dunn Index as it is given has two problems as concerns present purposes.

First, it evaluates clusterings as a whole, not individual clusters. So consider a distribution which contains two poorly-defined, overlapping noise/MUA clusters and a third separate, well-defined cluster whose waveforms have the characteristic shape of action potentials. The standard Dunn Index for this clustering would be low on account of the poor separation between the noise clusters. But the electrophysiologist only cares about the separation between the waveforms cluster and the two noise clusters. Thus clustering indices should really be calculated individually per cluster.

Second, the within-cluster metric is for each cluster is calculated independently of the clusters with which it is being compared. Typically this would be done by calculating the standard deviation of the distribution of distances from point to center. This may seem reasonable but in fact clusters exist in a high-dimensional space and the variance can be quite different along different dimensions. So it seems to make more sense to relativize the calculation of standard deviation for cluster i to cluster j by projecting each point to the line defined by their respective centers, and calculating the SD of *that* distribution of distances.

So that's what's done here. For each cluster, variance is calculated in the relativized way just described, and the index is defined as the minimum distance/variance when ranging over clusters. This is a highly modified version of the Dunn index so I call it the M-index. The code for this is in the file dunn.cpp.

7. File Formats

Input:
Binary file formats:

- *.apm (FHC)
- *M (REX/MEX)
- *.nev (Neuroshare)
- *.plx (Plexon)

WaveSorter also reads in tab-separated ASCII text files. The format of these is as follows:

<number of waveforms> *<samples per waveform>* [*et*]
<electrode #> *<timestamp>* *sample0* *sample1* *...* *sampleN*
...

In the first row, 'e' stands for electrode and 't' for timestamp. Including this data is optional, and either/both letters indicate that it is included. If it is, than that data value should occupy the corresponding column in subsequent rows. All subsequent rows contain the samples comprising a single waveform.

Output:
WaveSorter outputs data in a variety of formats. Note that for file names of the form *.*xxx*.*, the first asterisk is replaced by the name of the file which generated the output, and the second asterisk is replaced by electrode/channel number.

- REX E-file: REX users only. The user selects an E-file--corresponding to the session during which the M-file was generated, obviously--and WaveSorter creates a copy of this file with the old accepted waveforms having been replaced by the new ones. The file name is *_S0_E, instead of *E. The original E-file is not modified

- Cluster file (*.clu.*): This is a text file in the following format:

  <number of waveforms>
  cluster0 *cluster1* *...*
  <waveform 0 cluster>
  <waveform 1 cluster>
  ...

  So, the first row is the number of waveforms; the second gives all the clusters in the clustering, and then each cluster associated with the given waveform follows.

- Simple Binary (*.bin.*): This begins with a 4-byte integer holding the number of waveforms, followed by the bitwise sequence of accepted (1) or unaccepted (0) waveforms. Any leftover bits in the last byte are zeroed.

- Simple Ascii (*.ascii.*): Same thing as Simple Binary except that the output is in ASCII format, one record per row.
- Data file ("Save All"), *.data.*: A tab-separated file with the following format:

    1. First row: Number of waveforms, number of samples per waveform
    2. Second row: column headers describing values of subsequent rows. Accepted/unaccepted, cluster number, the values of all metrics and the value of each sample from the corresponding transform are all saved.
    3. Subsequent rows: One row per waveform, with the data as described in 2).

8. Citation and License

Citation:
If you use WaveSorter in published work, please cite it as

> Phillips, M.H. WaveSorter: A new free and open-source tool for the offline analysis and classification of neural waveforms. Program No. 508.12. 2012 Neuroscience Meeting Planner. New Orleans, LA: Society for Neuroscience, 2012. Online.

License:
WaveSorter is licensed under the GPL v. 3. See the *docs* folder for details. The basic idea of the license is to make legally rigorous and binding the concept of 'pay it forward'; you can do whatever you want with this software, but any derivative product you create that links to it or uses its source code you must also make freely available and similarly license under the GPL v. 3.

This license does **not** extend to the external clustering programs (AutoClass C, KlustaKwik, Kmeans++, Super-Paramagnetic Clustering) WaveSorter uses for automatic clustering. WaveSorter launches these, but is not in any way integrated with them; they are separate programs. These are governed by their own licenses, see the corresponding folder in *ClusProgs* for more information such as there is.